

## APPENDIX A: EXHIBITS

This appendix contains the documents referenced in the DSB Task Force Final Report. They include:

- A-1. Acceleration of Design and Development of Common Military Personnel/Pay Management Objective System - Resources Required (revised 8/31/96)
- A-2. Using COTS Software In Systems Development from National Research Council of Canada, Institute for Information Technology, Software Engineering Group (Copyright 1996)
- A-3. “Retool Human Resources” and “HR App Meets Critical Needs” articles from Datamation Journal, (6/15/95)
- A-4. Defense Science Board Military Personnel and Pay System Task Force Commercial Off The Shelf (COTS) Feature Comparison Charts (8/8/96)
- A-5. Ms. Padalino’s memo with SYBASE input to the Defense Science Board Task Force on Military Personnel Information Management Report - BAFO, (8/7/96)
- A-6. Mr. Selsor’s memo with GRCI input to the Defense Science Board Task Force on Military Personnel Information Management Report - BAFO, (8/7/96)
- A-7. BG Pellicci’s (USA, Ret) memo with ORACLE input to the Defense Science Board Task Force on Military Personnel Information Management Report - BAFO, (8/6/96)
- A-8. Mr. Larry Rinderknecht memo with EDS input to the Defense Science Board Task Force on Military Personnel Information Management Report - BAFO, (8/20/96)

# **Acceleration of Design and Development of Common Military Personnel/Pay Management Objective System**

## **Resources Required**

**QUESTION: What resources would be required to complete the functional definition and software development of an objective military personnel/pay management system with an IOC of 2001?**

This outline focuses only on this question. It does not address other critical factors, such as programmatic issues, management structure, and implementation. The estimates provided below are based on two staffing alternatives and the requirements definition approach currently in use by the Joint Working Group. Manpower estimates are only for definition of functional requirements and do not include personnel attached to the Executive Agents.\*

Background. At the Defense Science Board Task Force meeting on June 26, participants agreed that it should be both functionally and technically feasible to design and develop a common system (that would accommodate Service-specific requirements) with an IOC of 2001. It was recognized that this would require an acceleration of the current process for defining requirements and developing software. P&R was asked to provide an estimate of the resources required to complete these two tasks. It was also recognized that additional factors would need to be considered, such as the management structure, funding and reporting lines, Service-specific implementation plans, and the extent to which hardware currently being purchased by the Services could be used to deploy the objective system.

Assumptions. The requirements defined below are based on the following assumptions.

1. Military personnel management is the broad functional area defined by the 135 nodes of the process model developed by the Joint Working Group.
2. Resource-intensive case tools will continue to be used to define functional requirements.
3. The Joint Working Group experience that three months are required to complete functional definitions for each node, from initiation to hand-off to the software development team.
4. The Services will continue to be the primary drivers in development of requirements.
5. The Executive Agents (USN and USAF) for the prototype effort will continue as Executive Agents for the full development.
6. The accelerated program will require additional personnel for 2.5 years, beginning in October of 1996, after which a permanent, much smaller staff will remain in place for continued maintenance and updates.

Timeline. The timeline was developed working backwards from an IOC date of 2001. The proposed timeline is at chart 1.

Approach. The approach builds on and accelerates the work of the Joint Working Group and the P&R Information Management office.

1. Functional requirements will be incorporated modularly, with precedents.

2. Requirements will be analyzed concurrently rather than sequentially and passed to developers as modules are completed.

Staffing Alternatives. Two alternatives are presented for staffing the process for defining functional requirements. Both alternatives accelerate the current process from one module (or node) every three months to twelve modules every three months. Both alternatives assume significant contractor support, which is provided through the funding lines, and both alternatives assume continued support from the Information Management (IM) office or some comparable organization.

- Workshop approach: This alternative follows the pattern that the Joint Working Group has been using for requirements definition. Subject matter experts are brought in from the Services for two-week intensive workshops. Members of the JWG participate in each workshop to ensure continuity and consistency. JWG members also complete the complex processes required to complete the package for hand-off to the Executive Agents. This would require that, in addition to the part-time subject matter experts, the full-time staff would have to grow to about 55 personnel (including the IM staff) for the duration of the 2.5-year requirements definition period, allowing at least two JWG members to participate in each workshop and providing adequate support for integration and management.\*
- Team approach: This alternative creates standing teams for the 2.5-year period. Each team would focus on one module in each three-month period. The team members would be or become the functional experts and would obtain or provide whatever expertise was required. This alternative would require at least 90 to 100 government personnel (military and civilian) in addition to the contractor support.

Funds. Chart 2 estimates funds required to provide contractor support to complete the definition of functional requirements and to complete the software development by the Executive Agents. The estimates for the EA funding requirements was provided by the USN and USAF Executive Agents.

#### Pros and Cons for each Staffing Alternative.

##### Workshop Approach:

- Pros: Requires fewer full-time personnel assigned from Services.  
Ensure maximum participation from subject matter experts.
- Cons: Provides less continuity and creates new learning curves with each node.

##### Team Approach:

- Pros: Provides continuity and maximizes economies of scale.
- Cons: Requires more full-time personnel assigned from Services.

**\* To perform life-cycle management and coordination activities for the objective system, additional permanent staff of five will be required, for a total of 60 for 2 1/2 years, reduced to 25 thereafter.**

# MILITARY PERSONNEL MANAGEMENT OBJECTIVE SYSTEM

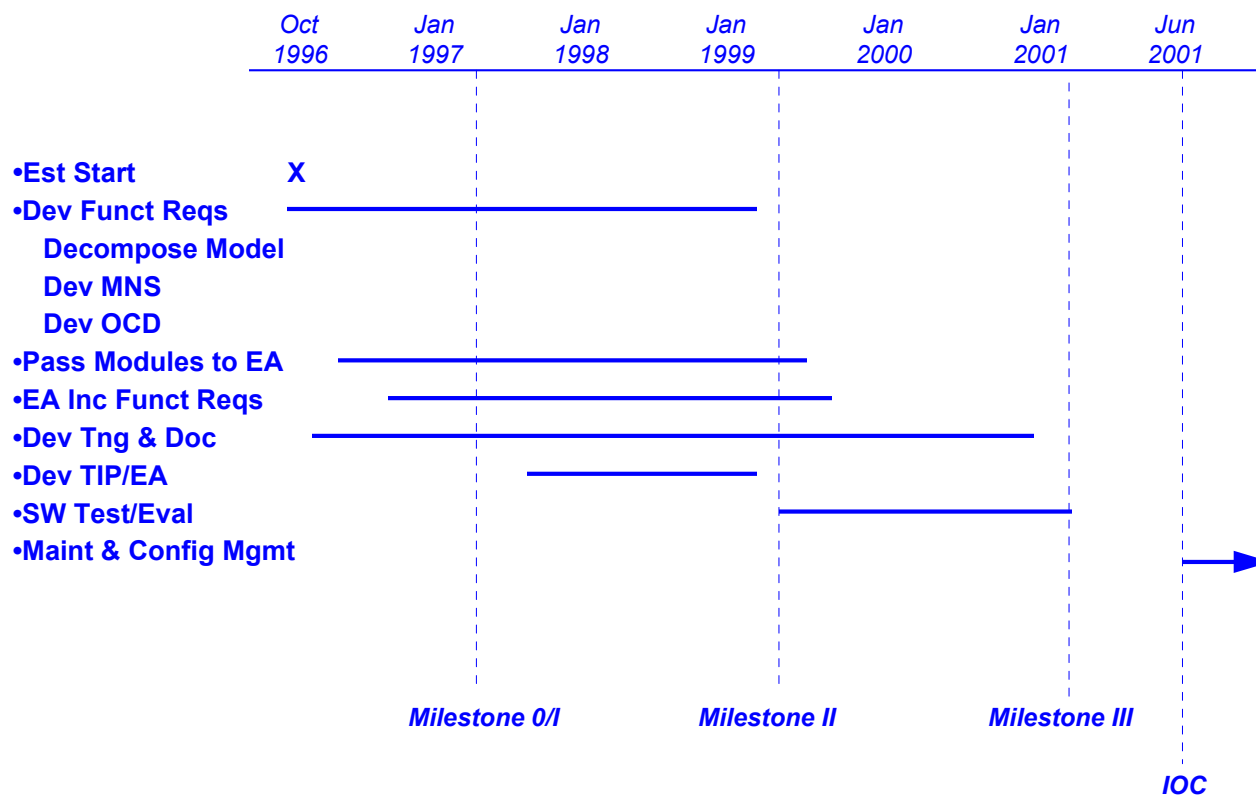


Chart 1

# *MILITARY PERSONNEL MANAGEMENT OBJECTIVE SYSTEM*

## *FUNDING REQUIREMENTS (\$000)*

	<i>FY97<sup>1</sup></i>	<i>FY98<sup>2</sup></i>	<i>FY99<sup>2</sup></i>	<i>FY00<sup>2</sup></i>	<i>FY01<sup>2</sup></i>	<i>FY02<sup>2</sup></i>
<i>Funct Reqs</i>	\$9,360	\$9,360	\$7,400	\$6,440	\$3,480	\$1,080
<i>EA</i>	\$11,760	\$6,280	\$8,100	\$8,100	\$8,100	\$4,200
<i>TOTAL</i>	\$21,120	\$15,640	\$15,500	\$14,540	\$11,580	\$5,280
<i>Available \$</i>	8,000	\$13,000	\$13,000	\$ 8,000	\$ 8,000	\$ 8,000
<i>Delta</i>	-\$13,120	-\$2,640	-\$2,500	-\$6,540	-\$3,580	\$2,720

*1 - Funding available per FY96 PBD*

*2 - Funding Contingent upon approval of OUSD(P&R) POM Issue*

*Estimated Program Costs      \$83,660*

Chart 2

**Using COTS Software In Systems  
Development from National Research  
Council of Canada, Institute for Information  
Technology, Software Engineering Group  
(Copyright 1996)**



## Using COTS Software in Systems Development

Using "Commercial Off-the-shelf" (COTS) software components to build systems has been proposed as a means of developing software with reduced risk and cost while increasing functionality and capability of the system. Building a system based on COTS components involves buying a set of pre-existing, proven components, building extensions to satisfy local requirements, and gluing the components together. The advantages claimed are that the COTS components are honed in the competitive marketplace resulting in increased capability, reliability, and functionality for the end user over what would be available from custom built components. COTS software components from different vendors are expected to be integrated easily, work in a wide range of environments, and support extensions and tailoring to local requirements.

The reality of the situation is quite different. Many organizations find that using COTS software carries a high risk and expense during development, during deployment, and during the ongoing evolution and maintenance of the system. Using COTS components, systems are often hard to build, to support, and to maintain. The problems encountered may be related to the processes an organization uses to build systems, in the technologies used to construct the system, and in the way systems containing COTS components evolve.

The Institute for Information Technology (IIT) is undertaking a research project to understand the problems associated with COTS software based development from the perspective of an organization that is trying to use COTS components to build systems. Given that developers from different organizations are going to continue to develop software that does not glue together easily, that may not evolve according to the users' wishes, that is continually evolving and changing, and that may not be properly supported, how should an organization go about using COTS software components for system construction? The objective of this research is to:

- Determine qualitatively the advantages and disadvantages of using COTS software in the construction of systems.
- Identify the types of systems that can benefit from the use of COTS components and the types of COTS components that can be used within these systems.
- Criteria for evaluating COTS software components.
- Modify existing software and system development processes in order to maximize the effective use of COTS software.
- Investigate technologies and architectures that enable the use of COTS software.

This research is sponsored by the Chief, Research and Development of the Department of National Defence .



National Research  
Council Canada

Conseil national  
de recherches Canada

Institute for  
Information Technology

Institut de technologie  
de l'information

---

**NRC - CNRC**

---

***COTS Software Integration:  
State of the art***

Mark R. Vigder  
W. Morven Gentleman  
John Dean

Software Engineering Group  
January 1996

---

**Canada**

Copyright © 1996 by National Research Council of Canada

Permission is granted to quote short excerpts and to reproduce figures and tables from this report, provided that the source of such material is fully acknowledged.

Copyright © 1996 par Conseil national de recherches du Canada

Il est permis de citer de courts extraits et de reproduire des figures ou tableaux du présent rapport, à condition d'en identifier clairement la source.

## TABLE OF CONTENTS

<b>1. INTRODUCTION.....</b>	<b>1</b>
<b>2. CHARACTERISTICS OF OFF-THE-SHELF BASED SYSTEM DEVELOPMENT... </b>	<b>1</b>
<b>3. OFF-THE-SHELF COMPONENTS: STATE OF THE ART .....</b>	<b>2</b>
3.1 CURRENT PRACTICES.....	3
3.1.1 Buy-and-adapt System .....	3
3.1.2 Integrating components.....	6
<b>4. EXPERIMENTS WITH OPEN SCRIPTING ARCHITECTURE (OSA) .....</b>	<b>8</b>
<b>5. RELATED WORK .....</b>	<b>10</b>
<b>6. ISSUES WITH COTS SOFTWARE INTEGRATION.....</b>	<b>12</b>
6.1 PROCUREMENT/DEVELOPMENT PROCESS.....	12
6.2 UNDERSTANDING AND EVALUATING COMPONENTS.....	14
6.3 EVOLUTION OF SOFTWARE.....	14
6.4 ARCHITECTURAL ISSUES .....	15
6.5 ROLE OF ARDS. ....	16
6.6 EMBEDDED SYSTEMS.....	16
<b>7. FUTURE RESEARCH.....</b>	<b>17</b>

1/25/96

## **1. Introduction**

The Software Engineering Group of the National Research Council (NRC) is currently undertaking a research project into the implications of using off-the-shelf (OTS) software to build long-lived military systems. The purpose of this project is to:

- Determine qualitatively the advantages and disadvantages of using OTS software in the construction of systems.
- Identify the types of systems that can benefit from the use of OTS components and the types of OTS components that can be used within these systems.
- Develop criteria for evaluating OTS components.
- Identify problems with the existing software and system development and procurement process that inhibit the effective use of OTS software.
- Investigate technologies and architectures that enable the use of OTS software.

This paper presents some initial results and observations regarding the OTS software usage. These results are based on the following activities:

- A series of interviews was conducted with personnel within DND who are involved in procuring or maintaining systems that contain off-the-shelf components.
- Round table discussions and interviews with researchers and commercial software developers who are interested in component based software.
- Review of the literature to determine the current state of the practice and current state of the research in building systems from OTS components, building software components, and open standards that enable the use of off-the-shelf software.
- Experimentation with different technologies and standards that are designed to enable the use of off-the-shelf components.

## **2. Characteristics of off-the-shelf based system development**

Building systems from off-the-shelf software components is an instance of a type of software re-use. It differs from other forms of re-use in that the system developer buys the components (usually without the source) from third party

developers and then integrates the components into the system. Characteristics of this approach to system development include:

- A component software product is designed to be sold in many copies to multiple customers with minimal changes.
- Pre-existence is one virtue - not only because it can shorten delivery schedules but because it means the customer can use pilot studies to rethink "requirements" and to investigate deployment problems.
- Honing in the marketplace, especially a competitive marketplace, improves specification, design, and implementation in ways that a waterfall development cannot anticipate.
- Vendor is responsible for ongoing support and maintenance - but this implies customer must accept upgrades
- No single customer has control over specification, schedule, or evolution.
- The specialized nature of the OTS product allows the customer indirect use of the rare skills of designers and implementers.
- Access to source code is unusual.
- The development process used may not be your favorite nor appropriate for easy configuration management and control.
- Internal documentation may be non-existent or not accessible.
- Technical information, especially as to limitations, performance, or resource consumption, may never have been collected.
- User level documentation, customer documentation, and training may be well developed.

Depending on the source of the components, they are referred to as Commercial off-the-shelf (COTS), Military off-the-shelf (MOTS), Government off-the-shelf (GOTS), etc.

### **3. Off-the-shelf components: state of the art**

As part of this research we have talked to numerous military, government, academic, and industrial people involved in the development of COTS components, or building systems from COTS components, and surveyed the literature to identify problems other organizations had had when building systems from COTS components [BRO95,NAS95]. The purpose of this survey was to understand the current practices of DND in terms of COTS software, determine the experiences and attitude of those responsible for software systems, and to

learn what kinds of problems (and solutions) people had experienced in relation to COTS software integration.

Within DND we surveyed eleven different groups or organizations and eighteen different individuals involved in about fifteen projects. The projects were primarily Command and Control systems or information systems.

There was no attempt to gather quantitative data. Very few useful metrics were available and it is unlikely that these would have any relevance across different projects. Thus the data gathered is qualitative and anecdotal.

### **3.1 Current practices**

Within DND all the systems looked at as part of this study were either information systems or command and control systems. We did not look at weapons systems but are aware of a number of systems that use OTS software, particularly where it is embedded within specialized OTS hardware.

There are different ways of using off-the-shelf software. The primary approaches we found within DND are the following:

- Buy and adapt. The buy-and-adapt model is characterized by acquiring a single complete working system that satisfies most of the requirements of the acquisition agency and adapting and extending it for local needs. The adaptation of the system is done by extending it through add-ons, interfacing with other applications, or modifying the off-the-shelf application through source code changes (But then is it really "off-the-shelf"?)
- Component integration. The component integration model of software development builds software systems by integrating a number of off-the-shelf components where each component satisfies some of the requirements of the system. This model usually depends on the use of some "gluing technology", which may be unrelated to the components, to provide an interface between components.

#### **3. 1. 1 Buy-and-adapt System**

The buy-and-adapt model of OTS reuse builds systems by purchasing an application which satisfies most of the system requirements and then extending and tailoring the application to satisfy local requirements. This model of development was used for both command and control and information systems. The applications bought were both military off-the-shelf (MOTS) and commercial off-the-shelf (COTS).

Within DND we observed two methods for modifying and extending systems:

- *API's*. Most of the systems which were bought and adapted had some kind of an API. The developer can write a controlling program that calls the

COTS component API as required. Typically this involves writing a "wrapper" around the OTS component to isolate the workarounds and extensions and provide a somewhat higher level of abstraction to the component's interface. The developer then writes the main program that controls the sequence of execution including calls to the OTS component (Figure 1a).

- *Modifying source.* If an OTS system does not satisfy all requirements then the supplier (or a contractor with access to the source code) can modify the application to satisfy the local requirements. However, once modifications to the source code is done the acquisition agency no longer has an off-the-shelf component but rather has a one-of-a-kind system. Using this approach there is a risk that these changes will become orphaned and the vendor will not support them during the normal course of upgrading the product.

The problems we have seen with systems that have been bought and adapted are:

- Limited source of supply. One argument for using COTS is that competition between vendors will drive prices down and improve quality of the systems. For many MOTS software applications there is limited choice of systems for purchase with minimal ongoing competition and these arguments do not apply.
- New releases of OTS component. Replacing older versions of COTS software with newer releases is difficult. New requirements or new hardware may preclude continued usage of the older version. Extensions and modifications made to the previous version must be re-integrated into the newer system. We saw three approaches to this problem:
  - Assume (hope?) the API and data formats of the new releases will not change significantly.
  - If extensions are added to the OTS software, have the vendor certify the changes as being compatible with all future releases.
  - If modifications have been made to the original source code, contract with the original developer (or someone with access rights to source) to make modifications to new releases. There is potentially a high risk (and high expense) associated with this approach: a complete process of analysis, development, documentation, and testing must be performed for the one instance of the product.

Within the commercial world the use of API's and source code modification are used but there is also a large amount of interest in using a concept known as



**frameworks.** A framework is a large scale component or an application which is designed to be extensible and integrated with other frameworks. With a framework one is not buying a component called through an API, but rather an implemented architecture inside of which one can embed extensions using techniques such as plug-ins and inheritance. The difference between traditional API based components and frameworks is shown graphically in Figure 1. In the traditional model (Figure 1a) the custom code defines the architecture and the COTS component is embedded in and called by the custom code; in frameworks (Figure 1b), localizations and extensions are embedded inside the framework, use the frameworks architecture, and are called by the framework.

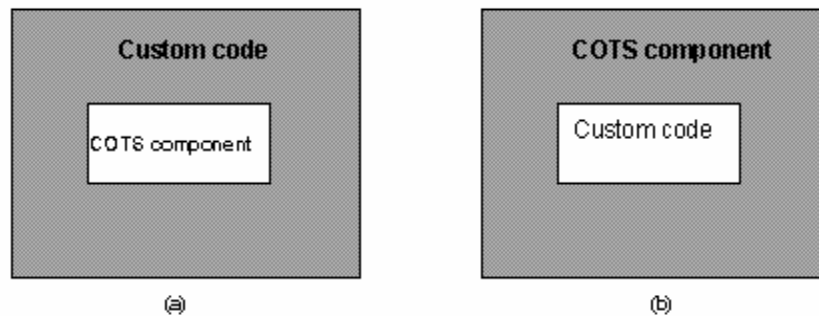


Figure 1.  
 (a) Traditional model: COTS embedded inside custom code.  
 (b) Custom code embedded inside COTS.

Frameworks can be constructed to provide generic services (such as the OpenDoc framework which provides an open architecture) or they can provide services within a specialized application domain, such as to the financial services industry, manufacturing, health services, etc.

A framework can be tailored and extended in a number of ways:

- **Plug-ins.** Developers can add functionality to an application by writing a "plug-in". A plug-in notifies the framework of its capabilities and services and the framework calls the plug-in as required. Effectively this reverses the traditional role of a component and the custom code: rather than the custom code calling the component through an API, the framework calls the custom code that is implemented as a plug-in.
- **Scripting.** A script is an executable fragment of code which is dynamically linked to components of the system. A script can be used to extend the behaviour of a component (by having the component execute the script), or it can be used as a coordination mechanism to integrate two or more components (by providing the "glue" for linking the components together). Over the last few years, numerous languages designed specifically as scripting languages have been developed and are being used on a

commercial basis (e.g., ObjectREXX, Visual Basic, AppleScript, JavaScript, tcl, Perl, Python, etc.) Currently there are two rival scripting architectures which are competing for market dominance: the Open Scripting Architecture (OSA) which is supported by IBM, Apple, and most of the computer manufacturers; and OLE Automation which is supported by Microsoft.

- Inheritance. Inheritance allows specific parts within a component to be specialized and modified. Current object-oriented standards allow inheritance to be used by developers who only have access to executable binaries and not to the source code. In contrast to scripting, which is often done by end-users or their support staff, inheritance requires a deep knowledge of the architecture of the framework and would normally be applied only by professional developers. Examples of this approach can be found in the CORBA object model and DSOM from IBM.

Within the commercial world building systems from frameworks can be viewed as a "just-in-time" programming model where components move along an assembly line from the developer through the integrator to the end user, and functionality is added at the latest possible point along this line. Component developers build large-scale components that provide services that appeal to a large market; integrators extend and combine the components to build systems; and end users and their support staff tailor the system for local needs.

Although frameworks are being developed (or at least talked about) within many commercial application domains there is currently limited activity in the military domain. There is an initiative at DoD's ESC Software Center (described in [BRO95]) which is being developed along the line of frameworks. The objective is to define a set of product lines each with its own evolving architecture and continuing qualifying COTS products. A client approaching ESC for a solution will be sold one of these standard architectures tailored to the clients needs. By this means they intend to have the capability to deliver systems to their clients quickly and with very little development effort. The price they pay for the short delivery time and low cost is that they do not expect to be able to satisfy 100% of their clients requirements, but they do expect to satisfy enough of the requirements that they are considered a cost effective solution.

Another example of a framework based approach is the DIGEST GIS system being developed at DREV. The GIS system is the framework to which plug-ins can be added to extend functionality. The objective is for development of both the GIS framework and the plug-ins to be taken over by commercial developers.

### **3. 1. 2 Integrating components**

The integration approach to COTS usage involves the developer buying two or more separate COTS software packages and integrating them into a larger system.

A component to be integrated can be packaged in many different ways. Within the systems we studied we observed the following component types:

- *Procedural libraries.* Component is delivered as a set of library routines which are linked in at build time.
- *Legacy applications.* Application is part of an organizations structure and workflow and must be included as a component within the new system.
- *Off-the-shelf applications.* Component is delivered as a stand-alone application (which may or may not have open interfaces and data formats.) Integration can take different forms, such as API calls, shared data in standard formats, "screen-scraping", event passing, drag-and-drop, etc. The most common form of integration observed was the use of shared databases and shared files.
- *Tools.* Tools are a mechanism to automatically build source code. An example is a GUI builder. Tools typically work by having the developer describe the system using the tool's language. The tool then generates source code that can be compiled and linked with the other components of the system.
- *System services.* Operating Systems, databases, windowing systems, and device drivers are typically purchased today as COTS components, and perhaps even considered part of the hardware platform on which the system is to be built.

In addition to the above component package types, there are others ways of packaging components. Two of interest are the following:

- Frameworks as defined in the previous section.
- OLE Components. The OLE standard from Microsoft promotes a number of technologies which are intended to enable reusable components. These include OLE Custom Components (OCXs) and OLE Automation servers. At DREV there was a project being undertaken to package a GIS system as a set of OLE component which can be extended through plug-ins from commercial developers.

The successful uses of COTS components which we observed within DND were the following:

- GIS. Numerous GIS libraries exist and provide adequate functionality for many applications.

- GUI builders. GUI builders are tools which are becoming mature and indispensable.
- Office automation software. Software such as calendars, word processors, and spreadsheets are used within many systems. The interaction between these applications and other components of the system were done primarily through desktop features such as drag-and-drop and enclosing files within e-mail messages.
- E-mail and messaging systems.
- Databases. Databases are an accepted part of many systems and are almost always bought off-the-shelf.
- Operating systems, including low-level software such as device drivers, windowing systems, etc.

The above list is made up of applications and components which are mature and pervasive in a large number of systems. This maturity is a likely reason why they have been successfully marketed as COTS software components.

Outside of the mature areas listed above, project and maintenance managers were generally skeptical about the feasibility and benefits of using COTS components to construct systems. Applications were not generally designed for interworking together, the applications required extensive wrapping, and large amounts of functionality needed to be added to the applications. In the small number of examples where we found COTS applications outside the above domains being integrated, the attitude of the managers was that the exercise could not be considered entirely successful. This was due primarily to problems in integrating and extending the functionality of stand-alone applications that were not designed to be integrated.

The current mechanisms we observed for integrating COTS components were the following:

- Procedure calls. The COTS component is accessed by linking to a procedural interface. Examples include components that are packaged as a procedural library, applications with an API, and databases with an SQL interface.
- Desktop supported capabilities. Desktops provide limited capabilities for integrating components through features such as drag-and-drop, clipboards, cut-and-paste, etc. This is generally how office automation software was integrated.

- Data sharing. For applications that store data in a standard format integration can be accomplished by having components read and write each others data. The shared data can be stored in shared files or in a shared database.

Problems we observed with the integrating COTS components include:

- Stability and support from suppliers was lacking.
- Integrating new releases of COTS software was a labour intensive and error-prone task.
- Contractors were sometimes unfamiliar with COTS product.

#### **4. Experiments with Open Scripting Architecture (OSA)**

Traditionally, commercial software packages have been bought and used on an "as is" basis. Commercial applications are sold as executable packages that have limited functionality for adding new services to the application. Users have been able to install the packages and, except for some minor tailoring, have had to accept the existing functionality of the software application.

Recently, there has been an initiative to open up software applications to allow users and third party developers to extend the functionality of applications and to integrate applications from different vendors to provide more sophisticated services to users. This trend has led to a number of competing and complementary technologies which are intended to allow developers to package open and extensible applications which can be enhanced and integrated by users and third parties.

As part of the research project, we are experimenting with emerging technologies that enable the use of COTS software components. One set of experiments we conducted was to take commercial off-the-shelf applications and to extend their capabilities using the Open Scripting Architecture (OSA). The purpose of this experiment was to determine, from a user's perspective, how effective the OSA currently is for extending application functionality.

The Open Scripting Architecture (OSA) is one of the two scripting architectures competing within the marketplace (the other being OLE automation). OSA is currently under control of the CIL consortium. It was originally developed by Apple Computer and is an integral part of their System 7.5 operating system. OLE automation is part of the OLE standard from Microsoft.

OSA has been adopted as the scripting standard for OpenDoc and thus will be available on most platforms. Currently, OpenDoc, including OSA, has been released for Macintosh System 7, Windows, and OS/2. The architecture is not

dependent on a single scripting language. Thus on the Macintosh platform, OSA scripts can be developed in AppleScript, or with some difficulty, Usertalk or tcl. Under OS/2, OSA scripts can be developed in ObjectREXX.

Scripting architectures are designed to allow developers close to the end users (and end users themselves) to integrate and extend applications. They are designed primarily for integrating and tailoring applications rather than for major development work. Applications built according to the OSA standard can extend their behaviour by executing scripts, and can communicate with other elements of the system by exchanging events.

Among the lessons we learned from these experiments are the following:

- Determining behaviour of COTS software components is difficult. Documentation, no matter how well done, is insufficient for understanding the detailed behaviour of components. Other than reading the documentation, other techniques we used for exploring component behaviour included:
  - Experimentation. Understanding the characteristics of a component always required experimentation. The OSA provides a means for CASE tool developers to build environments for experimenting with component behaviour.
  - Examples. Most of the COTS components we integrated had a large user base. Examples and workarounds built up by this user base were invaluable for solving many problems we encountered. This reflects our experience with many popular commercial applications where a large degree of support for the application comes from the user base rather than from the supplier.
  - Browsers. The OSA defines a means by which applications can advertise their objects, data structures, etc. Good browsers help a developer in categorizing and locating relevant information in an applications interface.
  - Recordability. OSA defines a capability whereby events can be recorded and disassembled into an appropriate high-level language. By recording an applications behaviour and studying the program fragment generated we were able to learn about the characteristics of some applications.
- Extending the OSA architecture leads to conflicts between the extensions. The OSA defines a means of adding extensions to the scripting language. Many of these extensions are being developed and marketed by third

parties. Unfortunately, with the large number of extensions being developed naming conflicts are arising between the extensions.

- Performance is low. The OSA architecture is currently not viable for time-critical applications.
- Concurrency is not supported.

## **5. Related work**

There are a number of initiatives being undertaken by different organizations on using COTS software components. This section lists some of the initiatives which are relevant.

The Software Engineering Institute (SEI) has initiated a project to research the implications of using COTS software to build military systems. Their research has focused on two initiatives: a laboratory project to investigate the integration of COTS CASE tools [ZAR94]; and a two day workshop on the use of COTS in system integration [BRO95]. The workshop proceedings are particularly interesting having brought together industrial and military people to talk about COTS integration within the following frameworks: technical; commercial/business; system architecture; open systems; and acquisition regulations. The workshop identified current problems and future directions of research in COTS integration.

NASA is a strong advocate COTS software and has a number of workshops and initiatives in this area. Their main focus has been on cost and risk reduction through reuse of software components across different projects. Most of their published data has focused on the acquisition/development process needed to foster use of COTS components. They identify a number of successful examples of COTS component usage. In addition to the more common COTS applications of database and user interface tools, they provide other examples such as a mission that integrates three different COTS applications: telemetry and command processing, orbit prediction, and health and safety monitoring [NAS95].

Among the lessons they have learned on successful usage of COTS are the following [NAS95]:

"The following beliefs about COTS packages are questionable

- COTS package solutions are less risky
- I can buy and modify a COTS package more quickly than I can develop it
- There is a COTS package for my application
- The COTS package works because there are a lot of copies in a lot of other organizations
- The vendor will keep the COTS package current.
- Vendor literature is true.

The following beliefs are reality:

- Vendors over commit themselves
- Vendors don't supply all services
- The software may not meet the requirements.
- The software may not be easily modified.
- I have very little control over vendor quality and schedule.
- My organization may have to change to accommodate this COTS package.
- Support costs for modifications may be 20% of the cost of the modification per year

Some fundamental differences between COTS based development and custom development are:

- COTS based development may need infrastructure earlier to demonstrate and prove the COTS package
- The COTS package may dictate standards, architecture, and design
- The COTS package may influence work flow
- Picking the wrong COTS package may be more expensive than fixing problems in custom software
- Issue resolution processes need to be in place earlier to resolve COTS package issues
- Issue resolution processes may be more complicated because of the addition of the vendor and possible incompatibilities between the vendor's practices and yours."

A number of new standards are being developed, driven primarily by commercial interests, with the goal of making component based software development a reality. The standards of interest include:

- The CORBA standard [OMG95] from OMG (<http://www.omg.org/>). The OMG, through CORBA, is standardizing an object-oriented approach to re-usable components. The CORBA standard is done at three levels:
  - CORBA which standardizes how component interfaces are specified and method invocations.
  - CORBA services which standardizes services which are common across many applications. Examples of services include security, object life cycle, transaction processing, etc.
  - CORBA facilities which standardizes frameworks (i.e., reusable components) within a specific application domain, e.g., manufacturing, financial services, etc.

The CORBA and CORBAServices have been standardized by the OMG. CORBA facilities are being standardized by interest groups within OMG but the standards are not yet in place. Although CORBA compliant Object Request Brokers (ORBs) are available commercially, a market has not yet developed for components built to this standard.



- The Component Object Model (COM) from Microsoft. This provides similar functionality to the CORBA standard (but not CORBA services nor CORBA facilities.) The COM is an implemented standard to which components are being built. It is limited by the fact that in its current form it does not support distribution and works only under Microsoft Windows.
- The document and desktop level component standards of OLE (from Microsoft) and OpenDoc (from CIL). OLE and OpenDoc are currently implemented on a limited number of platforms.

## **6. Issues with COTS software integration**

This section identifies and discusses a number of issues related to building systems from COTS components. The issues discussed are:

- Procurement/development process
- Understanding and evaluating components
- Evolution of software
- Architectural issues Role of standards
- Embedded systems

### **6.1 Procurement/development process**

A number of participants commented that one of the key elements to successful use of off-the-shelf software is that to gain the benefits while minimizing the risk an organization must be willing to accept the software component as-is with minimal changes. If the organization is not willing to accept the capabilities and limitations of the software as they exist then many of the benefits of off-the-shelf software will not be realized. This conclusion has been reached not only by ourselves, but by a number of other organizations which have researched the use of OTS software. The Auditor General of Canada which audited a number of large-scale government systems development projects made the following comment while auditing a Transport Canada system that is heavily dependent on COTS software:

"While the acquisition and installation of a commercially available software package should have been the least complex of the systems reviewed, the extensive modification of the software ... significantly increased the project complexity. As well as adding complexity, such modification may make the implementation of new releases of the software more difficult and costly.... A significant portion of the savings come from limiting the extent of changes to the commercial software package." [AUD95]

Therefore, in order to realize the benefits of COTS software a procurement process must be in place that defines requirements according to what is available in the marketplace, and that is flexible enough to accept COTS solutions when they are proposed.

The issue of impediments to using OTS software in military/government procurements is addressed by NASA [NAS95] and widely discussed in the SEI workshop [BRO95]. Both organizations have emphasized the need to accept the capabilities and limitations of OTS software and to adapt operational requirements to the availability of the COTS components. Currently, rather than approaching procurement with the attitude "what exists off-the-shelf and how can I use it", the procurement process identifies strict requirements which either excludes the use of COTS components, or requires large modifications to COTS packages in order to satisfy the requirements.

"In the old process, system requirements drove capabilities. In the new process, capabilities will drive system requirements...it is not a requirement if you can't afford it"

A number of the procurement issues brought up by SEI workshop include:

- Overly specific requirements often preclude the use of COTS components.
- Use of MIL-SPECs preclude the use of COTS.
- Different development standards are needed for development based on COTS software.
- Allow contractors to submit alternative proposals which may violate some stipulations of the solicitation but may be technically superior due to the use of COTS.

New military standards and procedures [MIL498] recognize the need for developers to include consideration of COTS components during the development process. Paragraph 4.2.3 and Appendix B of the 498 standard provide specific guidelines for incorporating 'reusable software products' into software systems. These guidelines make it clear that unmodified COTS must be handled differently than modified COTS. In general, the standard requires that modified COTS be treated as if it were normal developmental software. This means that it must be fully documented and tested within the scope of the new system. Much of the standalone testing and documentation requirements have been eliminated for unmodified COTS. In this case the standard provides for the use of existing documentation and a proven performance record as verification of the COTS software.

## **6.2 Understanding and evaluating components**

A component which is worth buying is likely a complex piece of software. In order to use such a component effectively it is necessary to understand it at quite a deep level. Understanding the behaviour of complex components is an extremely difficult task for a number of reasons:

- The documentation is incomplete or wrong. Even if the OTS supplier is conscientious about documentation, a complex piece of software will not be fully and correctly documented. Even incomplete documentation may be so massive as to be incomprehensible to any but the most experienced users.
- The interface may be very complex. Many of the standard components being marketed (e.g., OLE, DCE) have APIs with hundreds of calls. Not even the people working on these systems know how each API call behaves or the effect of particular sequences of calls.
- There are bugs in the software.

There is no easy solution to understanding complex systems. Documentation must be available and used, but a deep understanding can only be gained through extensive experimentation. Trying to understand these components has been called "an experimental science without any laws of physics." One advantage of using a COTS application with a large user base is that much of the experimentation has been carried out and the knowledge is available in the user community.

Many of the problems encountered with integrating COTS components cannot be determined before integration begins. The problems are of such a nature that they only appear well into the integration process [GAR95]. This makes estimation of schedules and resource requirements extremely difficult where COTS components are used. Extensive evaluation of the COTS component will be required to ensure not only that the component has the functionality to perform the required tasks within the system, but also that the additional functionality inherent within the component does not interfere with the system. The cost estimates must take this evaluation process into account.

### **6.3 Evolution of software**

Systems are constantly changing; COTS components within systems are constantly changing. This evolution of systems and their components has an impact in a number of ways on the maintenance of systems.

All people we interviewed stated that there was a large amount of work required to integrate new releases of COTS software into their systems. For a system constructed from multiple COTS components, with each component having its own release schedule, the cost of integrating each new release of each component becomes prohibitive.

The difficulties arise for two reasons:

- In the new release of the COTS component there will likely be changes in the behaviour, interface, assumptions, performance, bug fixes etc.
- Specializations, extensions, and workarounds made to the older version of the COTS component which must be integrated into the new component. There is no concept of "compliance" as in mechanical systems where parts are not defined to fit exactly but rather within certain tolerances.

One approach to the problem of evolution is to assume that the system (or at least the COTS component within the system) will never be updated. This will not be practical in all cases. The software may be dependent on a particular platform that is no longer available or is being updated; operational requirements may change; or bug fixes in the new releases may be required.

Integrating a new release of COTS software requires a significant development effort. First, the new release of the COTS software must be evaluated to determine what has changed from the previous release. Second, the new component must be integrated into the system. This may involve adding or removing workarounds, adding new extensions to take account of new behaviour, updating documentation and training procedures, etc. Finally, the system must be tested and verified.

#### **6.4 Architectural issues**

An appropriate software architecture addresses three issues related to the use of COTS software components:

- Plug-and-play of components. An architecture should allow pre-built components to be quickly assembled into larger systems. Components should be replaceable with a minimum amount of effort.
- Sharing of components/knowledge across projects. Within DND, many projects are implemented using a "stovepipe" model where each project develops components through all layers of the system with very little re-use or sharing with other projects. Many projects within DND have similar functionality and requirements; an appropriate architecture would encourage sharing of components across projects to take advantage of this shared functionality.
- Building systems as components for larger systems. All systems built should assume that one day they will be the off-the-shelf component which will be integrated into a larger system. Systems currently being built within DND have limited capability to be integrated into other systems. While this is a highly desirable trait, it is obvious that commercial vendors will be reluctant to add increased complexity to their product (at increased

development cost) to provide interworking with other systems unless it can be shown that there will be a return on investment. If we assume that DND cannot effectively influence commercial vendors, then the viewpoint should shift to ensuring that the COTS “glue” is flexible enough to accommodate integration with future systems.

### **6.5 Role of standards**

There is clearly a role for standards in enabling the use of COTS software for building systems. If standards are defined correctly and if software developers build components which comply with the standards then this goes a long way towards making open systems a reality. There are already a large number of standards which have had an impact in building plug compatible software components. Examples of these include tcp/ip, Xwindows, and SQL. There are many other standards in many other domains that have the potential to have a similar impact. Some of these address directly the issue of interoperable components (e.g., OLE, CORBA, ODP, OpenDoc, etc.) while others are targeted at a specific functional domain.

Although standards are important it must be recognized that there are many problems that standards do not address. For example, regardless of how pervasive standards become, they do not in any way eliminate the need for evaluating and experimenting with COTS components or solve the problems associated with maintenance and evolution of systems containing COTS components.

Another issue that must be recognized by DND is that to gain the benefits of standards it must be willing to accept the de facto standards which have gained acceptance in the marketplace. DND does not have the market force to dictate standards and attempts to mandate standards will likely lead to the use of software that is not widely supported in the commercial world. The role of DND should be to monitor and understand the standards, and possibly to try and influence the standards definition process, but once a standard has been accepted in the commercial world, be it a de jure or a de facto standard, this standard must be accepted as is by DND.

### **6.6 Embedded systems**

Although using COTS components is usually associated with information systems there are instances where COTS components can be effectively applied to real-time and embedded systems. There are many examples of real-time systems which make some use of COTS components. At the SEI workshop, the claim was made that the Boeing 777 has 4 million lines of COTS software including Microsoft products. (It was also claimed that to Bill Gates, Boeing was too small a customer to meet with.)<sup>2</sup>

Areas where COTS software components do have impact on embedded systems include the following:

- Embedded software. When buying a specialized piece of COTS hardware there will almost always be software embedded in the equipment.
- Specialized expertise. If embedded systems require access to highly specialized expertise, this expertise may be more readily provided by COTS software rather than attempting to hire the appropriate experts.
- Operating systems, etc. Embedded systems will generally use commercial operating systems.
- Standard non-critical components. Many embedded systems will continue to contain standard components that can be bought off-the-shelf and integrated. These include for example report generators, GUIs, databases, etc.

Issues to be resolved include:

- Certifying components that are critical. If a COTS component is to be embedded in a critical system, a rigorous certification process must be performed on the component.
- Constructing firewalls between the critical system and uncertified COTS components.

## **7. Future research**

Possible future directions of research relating to COTS software integration include the following:

- Develop techniques for evaluating and experimenting with COTS components. Select a number of products and attempt to define or develop tools which could be applied to a COTS component to pre-determine the overall functionality of the component. This may lead to the definition of a classification system which would allow project groups to quickly create short-lists of appropriate software components for their particular system. Another approach could be to attempt to analyze and establish evaluation criteria to be used to determine the suitability of a COTS component within the framework of a specific system.
- Define and evaluate technologies and architectures that promote reuse across projects and allow applications to be integrated into larger systems. Issues which could be investigated include:
  - Coordination languages designed specifically for integrating and extending COTS applications.

- Component architectures, including scripting architectures and frameworks.
  - Architectures which facilitate component reuse across projects.
  - Experimentation and monitoring of standards (CORBA, OLE, etc. ...). Experimentation is required to fully understand the functionality of these standards and determine how they may (or may not) facilitate the use of COTS software.
- Develop a guidebook for project and maintenance managers that identifies issues associated with COTS development. Contents of the guidebook could include:
    - Overview of development process applicable when integrating COTS components.
    - Costs, benefits, and risks associated with COTS software reuse.
    - Lessons learned from previous experiences with COTS components.
    - Tools and techniques for integrating components.
  - Pick an application domain within DND and determine how current state-of-the-art commercial software satisfies the needs of DND and how DND might change its systems to take advantage of COTS within this domain.

1/25/96

[NAS95] NASA workshops:

<http://bolero.gsfc.nasa.gov/c600/workshops/sswssp4b.htm>

[http://www-isds.jpl.nasa.gov/isds/cwo's/cwo\\_23/pbd.htm](http://www-isds.jpl.nasa.gov/isds/cwo's/cwo_23/pbd.htm)

[ZAR94] P.F. Zarrella, A.W. Brown, "Replacing the Message Service Component in an Integration Framework", SEI Technical Report CMU/SEI-94-TR-17, Software Engineering Institute, Pittsburgh PA, 1994.

<http://www.sei.cmu.edu/products/publications/94.reports/94.tr.017.html>

[BRO95] "Proceedings of the SEI/MCC Symposium on the Use of COTS in Systems Integration", A.W. Brow, D.J. Carney, M.D. McFalls ed., SEI Special Report CMU/SEI-95-SR-007, Software Engineering Institute, Pittsburgh PA, 1995. <http://www.sei.cmu.edu/products/publications/95.reports/95.sr.007.html>

[MIL498] MIL-STD-498, "Software Development and Documentation" <http://sw-eng.falls-church.va.us/AdaIC/standards/mil-std-498/>

[OMG95] "The Common Object Request Broker: Architecture and Specification, Revision 2.0", Object Management Group, Framingham, MA, 1995.

[AUD95]"Report of the Auditor General of Canada to the House of Commons. Chapter 12, Systems under Development: Managing the Risks", October 1995, Supply and Services Canada.

[GAR95] D. Garlan, R. Allen, J. Ockerbloom, "Architectural Mismatch or Why It's Hard to Build Systems out of Existing Parts", in 17th International Conference on Software Engineering, pp179-185, ACM, 1995.



**“Retool Human Resources” and “HR App  
Meets Critical Needs” articles from  
Datamation Journal, (6/15/95)**

## RETOOL HUMAN RESOURCES

*You bet payroll counts. Back-office human resource processing, too. But the real reason to go to a client/server HRMS system is to give line managers the power they need.*

By Lee The

**In lots of companies, "People are often treated worse than the equipment," says Bob Snelling Jr., senior VP of IS at Snelling International, a temp employment placement agency in Dallas. "They deserve to be treated with more respect."**

A good client/server HRMS suite can help companies do this.

Too touchy-feely a reason? No. It makes good business sense--and it's the right thing to do. The most concrete way to show respect isn't commemorating birthdays in the company newsletter but by letting people manage their own resources. Client/server HR systems could allow employees to get information about benefits or change personal data (new address, marital status, withholding, and the like). Managers can use those systems to do employee reviews or initiate hiring and disciplinary processes (all automatically routed to the right sign-offs). In short, client/server HRMSs can distribute human resource functions out to the line managers and to individual employees.

And, just in case you hadn't noticed, enterprises are getting squished. The winds of change have blown away layers of the hierarchy that used to provide the rungs on a career ladder. Employee evaluation, compensation, and career-planning issues have gotten much more complex and need to be administered much closer to the involved employees. All this doesn't just encourage client/server HRMS. It mandates it.

### RETOOL HR AND THE COMPANY

Jim Holincheck, a Chicago-based manager in Andersen Consulting's software intelligence group, says many of his clients are using HR systems to retool their workforce and shift to performance management. That shift includes everything from building appropriate skills in the workforce to compensating appropriately in the market.

You'll need to be able to tie into your skills database to see what has been planned and completed. This is one place where workflow comes in. Dun & Bradstreet has built proprietary workflow functionality into all of its products, including human resources. D&B is the clear leader here. Its HR Stream is fully integrated with D&B apps, so it sends around the actual transaction or piece of work, not just a message or form. SAP and Ramco say they will offer equally embedded workflow starting late this year. Also check out Edify. A number of vendors use Edify's workflow automation products in conjunction with Lotus Notes to help define workflows for certain processes.

**Tied to workflow is the fact that enterprise solutions are about providing the right information to the right people, regardless of departmental boundaries.** Therefore,

your HR system needs to be able to work across boundaries. Dowdy image notwithstanding, HR has been at the forefront of a lot of avant-garde technology, including scanning, imaging, information kiosks, and voice response. "We're seeing a lot of innovation in HR because it can help flow to the top line," Holincheck says. Clerical pencil pushing is being replaced by table-driven packages tightly integrated with payroll. "So, when changes are made to benefits elections, the payrolls are automatically adjusted."

### Bottom Line

## **THINK LOOOOOOONG TERM**

The HR system you buy needs to be more than client/server. It needs to be far more than an automated back-office HR clerk and paymaster. It must be able to reflect any and all organizational changes that your company may go through over the next decade. It needs an open architecture so you can incorporate voice response, telephony, and other new hardware/software technologies as needed. And the vendor needs to show a high level of consciousness about such trends.

Look for the ability to extend applications without disturbing the core functionality, and then couple that with sophisticated upgrade management. You need to be able to locate processing at the most logical place relative to performance and security. Object-oriented architecture best supports this, though few products are coded this way yet.

**Traditional HR/payroll packages are long on record keeping and short on reporting, so strong query and reporting tools are a must. And note whether those reports and queries can go against the operational data without needing a warehouse. You may see your HR department getting distributed; regional people may handle local data entry, recruitment support, and other tasks while headquarters does payroll and pension administration. Make sure your HRMS supports such application distribution.**

One zone of contention common to all client/server products is database support. Vendors like D&B, Humanic Design, and Oracle currently support just one database. Oracle HRMS (not surprisingly) only supports Oracle. The same is true for Humanic Design's Empire/SQL. D&B's HR Stream only supports Sybase. Others, like PeopleSoft HRMS, support several databases but forgo stored procedures and triggers and other deep-level implementations. This is a tough call, with advantages both ways. D&B is working on adding Oracle support, but it seems to be taking forever to do the port. Integral claims that its ground-up HR line rewrite, due this fall, supports Informix, Oracle, Sybase, stored procedures, triggers, and all.

**Deep database support is most necessary for OLTP, so even if you lean that way in other areas it's probably not as crucial with HR. Ramco, which calls itself a Microsoft shop, does its database access through ODBC. Holincheck has this tip: Payroll is the most likely bottleneck. If you're unsure whether or not payroll will fit**

**within your batch window, give the vendor 60 days to benchmark payroll in house before you sign on the dotted line.**

Frequent on-line operations do loom large in HR's future, starting with on-line time entry for employees. This can be sent by e-mail or modem to a central-processing operation, with links to attendance-monitoring and perhaps more general project management and labor allocation modules. There could be automatic feeds into payroll and even the general ledger. That way, the financials can use the hours reported, multiplied by standard rates, to get a cut at costs right away. Then they get trued up to payroll later.

Customizability is going to be key now and in the future. The trouble is, some vendors rely on proprietary toolsets. SAP's ABAP 4GL is more arcane than PeopleSoft's PeopleTools. D&B, Humanic Design, and Oracle use general-purpose development tools, which your shop may know already. But don't let that keep you from looking at PeopleTools to see what application-specific functionality can do for you. Whoever makes the tools, make sure the vendor shares your goals. If you want to implement workflow and the vendor doesn't, you may be out of luck. Also, vendor perceptions of how much you'll need to use a toolset vary. Oracle claims to deliver 90% complete HR apps, whereas PeopleSoft expects you to customize a lot more.

One good way to determine which products fit your needs-or how much customizing you'll need to do to make them fit-is to put them through a specific business scenario. One scenario Andersen's Holincheck has used starts with identifying an open position. Next, you search for internal and external candidates on file, place an ad in a number of newspapers, capture the cost, and tie it back to the requisition. Then, you scan in r, sum, s sent in response, and OCR them. You schedule interviews, record results, automatically generate an offer letter, scan in the acceptance letter, then at the point of hire transfer all germane data to the employee record. This is a common process. The trick is to see how well it works and how easily it flows from one stage to the next.

Long-time HR software vendors haven't all embraced the move out of that cozy little HR office, so a mainframe HR background is no guarantee that the vendor knows what's happening today. Likewise, some financials vendors have added HR products mainly to round out their product lines. In that case, you'd better find out just how well the vendor supports its HR line, from dedicated developers to in-house consultants.

Look closely at how much integration you really need between HR/payroll and financials. Being able to get it all from one vendor offers obvious advantages, but they're not as great as they might be from integrating other areas of the business.

## **PARTS AND PROMISES**

Just a year ago, it would have been simple to talk about client/server integrated human resource products for the enterprise. PeopleSoft was shipping a full suite. Everyone else had parts and promises. The vendors that have done the best job of playing catch-up

include Cyborg, Dun & Bradstreet, Genesys, Integral, Lawson Software, Ross Systems, Personal Data Systems, and Software 2000. D&B, Lawson, and Ross are best known as financials vendors. The others have been in the HR business for years, but not all of them have completed their client/server lines yet. Integral is missing a payroll module, and D&B is missing both payroll and benefits. JD Edwards has spent the last two years rewriting its in-house CASE tool, and the first products will ship later this year.

As the saying goes, "There's many a slip 'twixt the cup and the lip." That certainly applies to the challenge of converting from legacy to client/server apps, because there's often a world of difference between what some vendors say is possible and what really is possible. D&B, for example, provides a product that does the actual data conversion and claims it reduces conversion time by 70%. D&B also has a coexistence strategy so you can, say, run legacy payroll on the mainframe and tie it in with client/server HR. ADP even lets you tie its C/S HRMS to either a service bureau payroll system or an in-house one. And JD Edwards' new product line (code named "One World") will let you start out host-centric on your current hardware and operating system, then move to full distributed client/server processing incrementally.

Illustration by Daniel Pelavin

Columns | Features | Evaluation | Labs | Cutting Edge | Press Watch | In Box |  
DATAMATION

DATAMATION Copyright © 1995. All rights reserved.

## HR APP MEETS CRITICAL NEEDS

*When Snelling International went looking for a client/server HRMS package, it asked vendors to jump through a flaming hoop so hot only one package could handle it.*

By Lee The

**Snelling International's mission-critical application is--no fooling--its human resource management system. That's because SI's core business is hiring and deploying a huge, constantly shifting workforce of temp workers.** Few companies wring out their HRMS software like SI does. Yet despite having needs specific to the employment agency business, SI discovered that its ideal HR package didn't come from the vertical market software providers. To get what it needed, SI had to turn to a broad-spectrum HR package.

**To be fair, vertical HR packages for employment agencies are designed for companies that are smaller than SI. From its Dallas headquarters, SI operates through 251 franchise and company-owned offices spread across five countries. Last year, it processed over 50,000 W-2s and 338,000 paychecks, and sent reports to about a thousand local, state, and national regulatory agencies.**

SI's HRMS needs go way beyond the back-office complexities of administering a kaleidoscope of rotating jobs, employers, locations, industries, government regulations, and people. The boom in downsizing has spawned a concomitant boom in the demand for temporary and contract workers. Even when companies are looking for permanent employees, they are turning to "try before you buy" contracts, which let employers hire SI temps after 13 weeks.

Demand has outstripped supply. Thirty-thousand-odd placement agencies (several dozen of which operate on SI's scale) fight over good temp workers of every stripe. SI has been growing 30% a year, rising to \$300 million in revenue last year, but to maintain that growth in the face of such fierce competition, SI needs to take care of its workforce and manage its local agencies well. SI has no choice but to deliver superior HR management.

## Snelling International's IS Report Card

	<b>D&amp;B</b>	<b>Oracle</b>	<b>PeopleSoft</b>	<b>Ramco</b>	<b>SAP</b>
	<b>HRStream 3.0</b>	<b>Oracle HRMS 1.0</b>	<b>PeopleSoft HRMS 4.0</b>	<b>Marshal 1.0</b>	<b>R/3 2.2</b>
Returned requirements matrix	A	F	A	F	F
C/S payroll ready	Q4-95	Jul-95	A	C	Dec-95
Complex garnishments	F	F	A	F	F
Financials	B	C	B	B	B
HR	B	B	A	B	N/A
Language/toolset	A	B	A	B	C
Depth of database access	A	A	C	C	C
Tax	N/A	A	A	A	N/A
Industry penetration	D	B	A	F	D
References	B	B	A	A	B
Willing to fully demo products (including beta)	D	A	A	B	D
Quality of interaction w/vendor	B	A	A	D	F
<b>GPA</b>	<b>2.6</b>	<b>2.7</b>	<b>3.8</b>	<b>2.2</b>	<b>1.3</b>

## WEEKLY GRIND

That just wasn't going to happen with the legacy system, as is or upgraded. Back in 1989, when SI's temp operation was much smaller, the company bought a NetWare LAN-based accounting system from Platinum Software. Then it built a payroll system for it with PC Magic, a 4GL modified to incorporate Platinum's data structure. Everything worked in character mode or paper forms.

By 1994, the system was taking a week to grind out each week's payroll for an average of 6,500 workers. The programmers complained that Magic's architecture rivaled mainframe systems for rigidity. The system lacked an automatic interface between payroll and financials. The system couldn't consolidate automatically across business units. There was no provision for decision support. And payroll was so customized that SI couldn't keep up with Platinum updates. Nor did they see Platinum For SQL (Platinum's own upgrade product), as a possible solution. Platinum didn't offer payroll, and as it was refocusing on its core financials, the future was unclear for Platinum's HR package, too.

In late 1993, Bob Snelling Jr., senior VP of IS, and Buck Buchanan, VP of IS, teamed up to start looking for a better way. They wanted to know who had integrated HR/payroll/financials. They got an initial list of over 100 such vendors from the American Payroll Association. They got to work on the phone calling the vendors, perused HR trade magazines, checked out HR trade shows, and worked up an RFP.

Snelling says 60 of those vendors said they thought they had a product that fit SI's needs, but he and Buchanan quickly found themselves dealing seriously with only a handful of vendors. And "as we kept boring in on them, giving them more requirements, more stopped calling us or told us to go with others," says Snelling. At the behest of top management, Snelling and Buchanan had confined their search to employment agency verticals and were trying to find something that ran on the AS/400 or NetWare LAN they already had. These products were much cheaper than mainstream client/server products.

**Snelling and Buchanan both believed the company needed a modern client/ server package, but company executives weren't so sure, and they were used to making decisions and handing them off to IS to implement. Snelling needed to move IS out of the hired hands category before the company wound up with a system that wasn't actually new and wouldn't carry SI into the future properly.**

The change was so huge that Snelling knew a simple decree wouldn't work-even coming from a senior VP and son of the chairman. "I could have gone out by myself as senior VP of IS and made the decision," he says. "But I knew there would be tough times during conversion and installation. Then people would say, 'He doesn't know what he's doing' or 'He's getting kickbacks.' So we had to include everyone. That way, when the tough times come, they have to say 'Yes, this is tough. But there's no way around it.'"

**So Snelling did a Texas two-step. First, he included everyone from the chairman of the board to the staff accountants in payroll in each vendor evaluation. Second, he brought in a reputable consulting organization, CSC--not for specific product recommendations, but for strategic direction. "We had CSC educate upper management on the way technology was heading," says Snelling. CSC convinced them that SI had to go client/server and to reject any HRMS products that weren't. This was what Snelling and Buchanan had already told them for free, but it was money well spent.**

"My most difficult challenge with IS," Snelling says, "was to politely get the executives to understand that they didn't know what was best for the company in automation" and that IS did know. CSC helped Snelling accomplish both of these delicate tasks. It also addressed the issue of reautomating SI's headquarters in general. SI had multiple databases of information about franchises. If a franchise address changed, it required multiple manual updates. The need to update payroll and financials and acquire HR software had to be seen in this perspective, not as a set of disconnected issues. "We needed a consulting company to help save us from making multimillion-dollar mistakes," Snelling concludes.

## **CTRL-ALT-DEL**

After seeing eight vertical vendor demos and hearing out CSC, Snelling and Buchanan--and everyone else--whittled the list down to, well, zero. Just as Snelling and Buchanan had predicted, none of these products would do. They offered little more than payroll and couldn't handle the complexity and scale of SI's operations.

So they started over, this time looking for mainstream client/server HR products. They went back to the initial list of 100 and to the mainstream computer press, especially for information about new products, like Ramco's Marshal Human Resource Management.

All in all, SI talked to 29 companies on the second pass. Lots of vendors backed out fast because of the complexity and scale of SI's needs or because their suites of client/server



tools were incomplete. SI's requirements list had grown to over 247 line items, covering systems, security, data, check processing, payroll processing, taxes, error adjustments, reporting, human resources, accounts receivable, and workers' compensation. Other specific HR requirements included insurance; EEOC; new-hire tracking; tracking I-9s, W-4s, and W-5s; tracking vacations and holidays; benefits; training skills/experience; job/work experience; and resume information.

**SI needed to be able to track training, both for job skills and government-mandated safety training. And to compete for the best temps, SI needed to be able to offer benefits, which the HR system had to monitor and manage.**

The requirements list was intimidating enough. But the flaming hoop Snelling and Buchanan devised for the contenders came from the depths of payroll: garnishment management.

**Garnishments have been a nightmare for SI. A lot of temps have multiple garnishments from multiple counties--or even states--for everything from child support to loan repayments. Multiple legal judgments sometimes mandate dollar amounts that, taken together, exceed the temp's pay. If that happens, the software has to figure out how much goes to whom based on percentages determined in court judgments and limits as to how little you can leave an employee with.**

Rules vary from state to state, and you have to figure in reciprocity laws between the states when more than one is involved. A temp may live in one state and work in a second; the SI office may be in the first state, the second state, or even a third. To make things even worse, some temps try to beat the system by having extra withholding taken out to artificially depress their pay. The software needs to spot that and deduct garnishments before the extra withholdings.

**Snelling and Buchanan gave the vendors real-life garnishment scenarios to compute, involving multiple job assignments and figuring which and how much would come out for child support, home appliance payments, utility bills, Texas state taxes, and more.**

Buchanan says that a lot of vendors just didn't take these complexities into account when they designed their software. When confronted with SI's garnishment deductions test, several vendors said that the scenarios were ridiculous. Buchanan retorted that not only could he show them numerous real examples but SI's own homebrew package could handle them.

**And SI laid a trap for unsuspecting vendors. Remember those Texas state taxes that had to be figured in? Well, Texas doesn't have state taxes. "We were sneaky about it," Snelling laughs. He adds that most of the vendors made up a tax amount anyway, so the trap worked.**

About 18 vendors took the garnishment test: 10 vertical packages and eight client/server. All of the vertical packages failed miserably, others came closer, but not close enough. "A lot of software packages just didn't work right" on garnishments, Buchanan says, adding that Oracle's chief of research and development decided to revise Oracle's approach to the problem based on the results of the test. Computer Associates said its CA-HRISMA software could handle it, but failed to provide anything like the right amounts in several tries. In some cases, the software prompted for the amounts, then ignored them. Only PeopleSoft HRMS got the right answers and was able to handle interstate reciprocity laws.

#### What's so Client/Server about HRMS?

### **ULTIMATELY, NO CONTEST**

Buchanan says that PeopleSoft HRMS outshone its competitors elsewhere, as well. "I don't think any of the other vendors' packages were quite comparable," says Buchanan. "PeopleSoft had all the pieces and more effective tracking." The SI team loved all of the tables that came with it. In fact, according to Snelling, "the winner in everyone's mind from early on was PeopleSoft. The comments after every review of the product went "Boy, if we won the lottery, it would be great to have-but so expensive."

The serious finalists to emerge from SI's gauntlet were D&B HR Stream 3.0, Oracle's Oracle HRMS 1.0, PeopleSoft HRMS 4.0, Ramco's Marshal 1.0, and SAP R/3 2.2. And it turned out they were all in PeopleSoft's price bracket. After some soul-searching, the executive team decided to expand the buying budget. Funds also had to be allocated for upgraded hardware and the customization needed to adapt a broadband package to SI's specific needs. The most important change involved a standard procedure for a temp help agency: After SI pays a temp, it has to invoice the client, then pay the franchi--see after deducting SI's royalty and other moneys owed from each franchise's individual agreement. The verticals did all that, of course. So it meant that Snelling and Buchanan needed to look at each vendor's customization capabilities closely.

PeopleSoft uses a proprietary toolset called PeopleTools along with COBOL and C. Oracle and SAP use proprietary 4GLs. Ramco uses VC++, a 3GL. Buchanan compared PeopleTools to PowerBuilder (D&B's tool), and found it not as complete. But COBOL batch-processing support made up for that by not forcing everything to go through a screen interface. Buchanan's biggest worry about using the 4GLs in general was that they would make it easy for someone to go in and mess things up.

**SI also seriously considered the treatment it had received from the vendors. Some vendors put their cards on the table, while others treated SI's team like mushrooms; they showed off glossy color brochures that implied products in development were actually complete. Some were more interested in dishing dirt on the competition than in showing off their own stuff.**

Snelling did note that PeopleSoft's one-SQL-call-fits-all approach precluded the stored procedures and triggers needed for optimum performance. But he was resigned to having his staff code those in order to get PeopleSoft's other advantages. He also accepted the need to support it with fast hardware, figuring those costs into his calculations. Ultimately, he could accept slower performance when it was coupled to superior functionality and friendliness.

## **TELLING INSIGHT**

Reference checking and site visits were telling. "The first time we visited a PeopleSoft site, we thought we had somebody who maybe got paid under the table," Snelling said. "But every single site we talked to raved about the product and its implementation. They said yes, it's pricey, but worth it. The Oracle, SAP, and D&B sites said they liked it, it worked well. But they weren't raving."

If SI chose Oracle or Ramco, Snelling knew he'd be working as a beta site and to some extent as a codeveloper. He didn't want to chance it with Ramco, which was new to this country, but he did get close to doing that with Oracle, since SI had opted for Oracle's database. SI ultimately decided not to become a beta guinea pig.

**Snelling even had the brass to call SI's 25 chief competitors and ask them what they used. Surprisingly, "18 were quite open with me," he says. Three of them had already bought PeopleSoft, and Snelling feels this would give them all clout with PeopleSoft on issues common to their industry.**

The financials weren't ready by the time SI signed the contract in late December, but Snelling was confident that they'd be ready by the time SI was ready to install them. The fact that PeopleSoft had always been aboveboard about where it was on each piece helped build that confidence. However, Snelling would have bought PeopleSoft's HR/payroll package and someone else's financials if need be.

**Ordinarily, "a corporate headquarters with 140 employees is never in 100% agreement on anything," says Snelling. "When it comes to software, there are people who staunchly defend their favorites. But from the chairman who'll never have to use this software to IS programmers and system analysts to payroll tax department executives to managers and staff-it was a unanimous decision.**

# DEFENSE SCIENCE BOARD MILITARY PERSONNEL AND PAY SYSTEM TASK FORCE COMMERCIAL OFF THE SHELF (COTS) FEATURE COMPARISON CHARTS

These Feature Comparison Charts provide you a guide to look at competitive commercial off the shelf products (COTS) in your selection process. As you look at the competitive products Chart #1, you see that Product A provides you all of the features needed for Distributed Client/Server applications at 25% less cost. Chart #2 contains the summary advantages of your selection. Although these Feature Comparison Charts are provided as examples only, they contain the principal features required for an integrated single military pay and personnel system.

Feature Comparison Chart #1

Feature	A	B
<b>Global Issues:</b>		
Compiled code support	No	X
16 and 32-bit application creation		X
Full object-oriented support:		
- Class Libraries	X	X
- Encapsulation	No	X
- Function Overloading	X	X
- Multi-level inheritance	Limited	X
- Polymorphic Messaging	No	X
Platforms	Windows 3.1, NT, OSF/Motif, Macintosh, character mode.	Windows 95, Windows 3.x, NT, DEC Alpha NT, Mac, UNIX (Sun Solaris)
<b>Functional Capabilities:</b>		
Application Partitioning		X
C++ Class Builder	Limited: Can call DLLs, but cannot support the language	X
Class Library (Pre-built objects, sample framework & services via the Foundation Class library)	No	X
Central design repository for defining and storing extended attributes	X	X
Configuration Management	No	X
Database engine (included)	X	32-bit Sybase SQL Anywhere
Database stored procedure and trigger support	via database supported products	X
Database interoperability & scalability to Sybase SQL Server for full enterprise deployment	No	X
Database support	Oracle RDBMS (native access) and ODBC- only connectivity to non-Oracle databases	Native Access to: Oracle, Sybase SQL Server, BM DRDA, Informix SE, Informix Online, MDI Gateway for DB/2, Microsoft SQL Server, and ODBC connectivity to Btrieve, IBM DB2, DB2 for AS/400, dBase II III IV V, Excel, Netware SQL, Paradox, Text, Sybase SQL Anywhere
Data pipeline for data conversion & migration between databases	No	X
DDE	X	X
DLL support	Limited: can call, but not create.	X
Debugger (built-in)	X	X
Foundation Class Library (reusable, pre-built objects and services to accelerate development)	No	X
Help-online	X	X
Integrated Mail support (MAPI)	Via integration w/3rd party only	X
Library for Lotus Notes	Via integration w/3rd party only	X
Object browser (including OCXs)	No	X
Object management	X	X
ODBC version 2	X	X
OLE 2.0 custom control (OCX) Support	X	X
OLE 2.0 extended support		
- Plug & Play with any OCX	No	X
- OLE automation - OLE automation servers	No	X
- OLE server-enabled DataWindow	No	X
- Point & click SQL interface to OLE	X	X
Point-and-Click Technology that eliminates SQL coding	No	X

Feature Comparison Chart #1		
Feature	A	B
Risks:		X
(e.g. vendor bankruptcy, obsolete product)	Products favor a closed architecture that discourages customers who want to employ an open systems approach.	None
Pricing	\$3,995	\$2,995
Feature Comparison Chart #2		
	The Summary Advantage	
Requirement/Feature	A	B
Fast Applications		
Compiled Code	X	
Native Drivers	X	
Distributed Objects		
OO with Inheritance	X	
Partitioning	X	Stored Procedures Only
Data Anywhere		
DataWindow	X	
Full ODBC Support	X	X
Applications Anywhere		
Multi-Platform	X	X
Multi-Platform Controls	X	

**Ms. Padalino's memo with SYBASE input to the  
Defense Science Board Task Force on Military  
Personnel Information Management Report  
BAFO, (8/7/96)**



7-Aug-1996

OUSD(PNR) R&R-IM  
Attn: Norma St. Clair  
4015 Wilson Blvd.  
Suite 1212  
Arlington, Va. 22203

Subject: Criteria for selecting COTS software for DoD-wide Personnel  
and Payroll

Dear Ms. St. Clair:

Sybase is pleased to have the opportunity to provide this input on criteria for selecting a COTS application for the DoD's Personnel and Payroll requirements. Although Sybase does not directly produce software for Personnel and Payroll, our open database management, middleware, and application development tools provide the technology foundation for many of the industry's leading COTS application vendors. The criteria these COTS applications vendors used in selecting Sybase products as their foundation technology was driven by their customer's needs. It is that criteria described herein and also the criteria that will best help the DoD in selecting a COTS Personnel and Payroll application with the lowest life-cycle cost and shortest implementation time.

**Criterion 1 - Open Database:** The COTS application should run "natively" with multiple SQL RDBMSs, including either the big four-- e.g., Sybase, Informix, Oracle, Microsoft-- or, at a minimum, the two specified in the Defense Information Infrastructure (DII) Common Operating Environment (COE) published by DISA-- e.g., Sybase and Oracle. This means that the application can directly run on a choice of RDBMSs without any performance loss or redundant data storage requirement. COTS applications that require a proprietary vendor file system or specific RDBMS to natively store data and then redundantly copy data to other RDBMSs do not meet the Open Database criterion. Selecting a COTS Personnel and Payroll application should in no way force you to select or standardize on a single database.

Open Database support would benefit the DoD primarily through cost savings in two key areas: 1.) those accrued from competition in acquisition organizations need to purchase a new RDBMS to run the COTS application



and 2.) those associated with organizations being able to re-use their existing inventory of NAME and save on purchase and re-training costs. Secondly, adoption of the Open Database criterion would benefit the DoD through the inherent benefits from the flexibility provided by a choice of SQL RDBMSs--these included leverage over vendors to provide first rate support or risk swap out, and the ability for the DoD to select the best of breed RDBMS at the time of deploy given that title will continue to change hands quite regularly over the period of deployment.

**Criterion 2 - Open Tools:** The COTS application should be built upon and modifiable with open application development tools that support rapid application development, facilitate managed re-use, and are widely used in industry as well as the DoD. The Open Tools criterion requires that the COTS application toolset run "natively" with multiple SQL RDBMSs for the same reasons listed above for the Open Database criterion and for the additional reason that non-native interfaces, such as the Open Database Connectivity (ODBC), do not meet the performance requirements of large, complex applications like those of the DoD's Personnel and Payroll would applications.

The additional benefits of the DoD adopting the Open Tools criterion (i.e., over and above the benefits described under the Open Database criterion) are in two areas: 1) training and personnel cost savings and 2) in development time savings. By selecting a COTS application that employs a toolset that is widely used in industry and government, the DoD will find it much less costly to find, train, and acquire personnel resources with sufficient expertise in the required locations. The also speed up its development effort as it wouldn't have to spend time and moneys to train its staff or contractors in the use of vendor-proprietary toolsets. Finally, the DoD would benefit in cost and development time savings from the "rich and robust" third-party market for software development environments and productivity enhancements that are ever-present around industry-standard toolsets.

Sybase realizes that the DoD will need to consider many other criteria (such as functionality fit with DoD's requirements, vendor market share, ease of modification, etc.) in selecting a Personnel and Payroll application. The above described Open Database and Open Tools criteria will ensure the technological underpinnings of the selected Personnel and Payroll application are consistent with DoD's objectives for selecting COTS-- life-cycle cost savings and minimized implementation times.

Should you have questions, do not hesitate to contact me at 301/896-1757.  
Thank you for giving Sybase this opportunity to participate in this important  
Defense Science Board action.

Sincerely,

A handwritten signature in black ink, appearing to read 'Peggy Padalino', with a long horizontal flourish extending to the right.

**Peggy Padalino**  
**Account Manager**

**Mr. Selsor's memo with GRCI input to the  
Defense Science Board Task Force on Military  
Personnel Information Management  
Report - BAFO, (8/7/96)**



---

1900 Gallows Road Vienna, Virginia 22182 (703) 506-5000

August 7, 1996

Chairman, Defense Science Board  
c/o Norma J. St. Claire  
Office of the Under Secretary of Defense Personnel and Readiness  
4015 Wilson Blvd., Room 204  
Arlington, VA 22003

Reference: Input to the Military Personnel Information Management Task Force Report

The purpose of this memo is to provide observations regarding EC role of adapting a COTS solution for the MPM21 Objective System and the degree to which that solution may simplify design/development tasks, reduce timelines and save money.

Clearly, there are good analogies in the business community that demonstrate the effective use of COTS solutions for integrated personnel and payroll systems. It would be in the government's best interest to explore several of the most relevant solutions so that a full range of alternatives may be understood and valuable information gained from these industry experiences.

It should be noted, however, that the MPM21 Objective System will need to satisfy a demanding range of requirements, including:

- Integration of active, reserve, and reserve components,
- Ability to scale solutions so that they provide effective functionality to every operational level from detachment to top of the system,
- Ability to accommodate multiple hardware/software suite combinations at every level,
- Ability to integrate across personnel and pay functions as well as share data with other functional area systems (i.e., logistics, medical),
- Ability to accommodate wartime and peacetime operational environments,
- Ability to integrate with legacy systems, and to interoperate and share data with other enterprise systems on the DII.

Determining the ability of COTS products to address all of these requirements is critical to program planning. A gap analysis may show how much functionality comes "out of the box", but equally important is the complexity of functionality not addressed by COTS products. The tailoring and extension of COTS software to meet MPM21 requirements

Chairman, Defense Science Board  
August 7, 1996

may test the engineering rule-of-thumb, which is that it takes 90% of the development cycle to achieve the last 10% of critical functionality. Only a full laydown of essential requirements can lead to that determination. As a result, shortening of the timelines due to adapting a COTS-based solution may make it infeasible to provide a system that meets all of the stated requirements.

Additionally, a series of fundamental technical issues must be evaluated before a program timeline and life-cycle cost estimate can be established with reliability. These issues include: the degree of COTS tailoring needed; the amount of new software necessary to meet functionality not addressed by COTS; the complexity of integrating new modules with COTS products; performance trade-offs of running a hybrid COTS/designer software system on all hardware/software suites used by MPM21; and the maintenance to support modified COTS, new software, and integration software products created to support this project.

COTS packages often incorporate industry best practices and provide excellent functionality. Leveraging existing COTS software may shorten the application development portion of the system schedule, however, industry metrics show that less than 20% of project costs are expended developing code, while integration tasks can consume close to 25% (derived from case studies used in Checkpoint project costing software developed by Software Productivity Research, Inc.). Tailoring and integrating COTS-based applications into enterprise environments present unique management and technical issues distinct from traditional development efforts. Based on our experience participating in the development of other large DoD enterprise systems, such as, Joint Computer-Aided Acquisition and Logistics Support (JCALS) System, Reserve Component Automation System (RCAS), and the Defense Investigative Service (DIS), we have confirmed that the integration of COTS can be more complex than fairly well-defined application development segments and as a result, can have a significant impact on the system development schedule.

We agree that COTS solutions should be vigorously pursued for MPM21. By using proven COTS products to satisfy mission requirements, DoD will realize significant benefits by leveraging complete and tested products for its functional modules. Before enterprise-wide deployment, however, it is important to prototype and benchmark these COTS-based solutions. The purpose of this benchmarking effort is threefold. First, it can be used to update the objective system architecture by providing valuable information on the degree to which COTS software meets requirements and by identifying new software modules that are needed to realize full functionality. Second, it can be used to identify the full scope of system integration tasks and validate the COTS-integration process. Third, it will help define a standard method for implementing interfaces to legacy systems, as well as achieving interoperability and data sharing with core DII systems

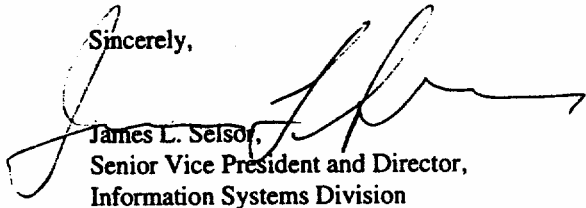
Chairman, Defense Science Board  
August 7, 1996

(i.e., GCSS). Accordingly, the initial COTS initiatives should be viewed as interim solutions that can be fully exercised by all services to validate the utility of the product and to help prioritize the addition of greater functionality. Lessons learned from these efforts can be incorporated into the final objective system through an aggressive product improvement program which gives the added benefit of reinforcing user confidence by incrementally proving the value of progressively enhanced system updates.

We understand that an incremental development approach may be more time consuming than originally envisioned. Lessons learned from past and current large-scale development efforts, however, demonstrate to us and our government clients that a progressive development plan offers the highest chance of success in fielding a system that responds to the complex requirements that an MPM21 objective system must address.

Thank you for the opportunity to provide this input. Building a DoD enterprise military personnel system demands the best industry can offer. We realize the magnitude of the challenge faced by the Defense Science Board and DoD in this endeavor and we would be happy to provide further details or respond to questions that might arise from our observations. If we can provide additional information please call Paul Schuessler at 506-5336.

Sincerely,



James L. Selsor,  
Senior Vice President and Director,  
Information Systems Division

**BG Pellicci's (USA,Ret) memo with ORACLE input to  
the Defense Science Board Task Force on Military  
Personnel Information Management Report - BAFO,  
(8/6/96)**



August 6, 1996

Ms. Norma St. Claire  
OSD, Personnel & Readiness  
4015 Wilson Blvd Ste 1212  
Arlington, VA 22203

Dear Norma:

On behalf of Oracle Government, I am pleased to provide our final observations and input concerning the Defense Science Board Task Force on Military Personnel Systems. Attached is our two(2) page input. Also, included is the clarification to question 8.

Regards,

A handwritten signature in black ink, appearing to read "Jack Pellicci", with a stylized flourish at the end.

Jack Pellicci  
Vice President  
Oracle Government

JP/rp

enclosure(s)



## **Recommendations to the Defense Science Board (DSB).**

As a leading developer and provider of information technology and services, Oracle would like to take the opportunity to advise the DSB in the area of COTS software. Oracle believes that the following evaluation points are a viable litmus test for any vendor under consideration to fulfill DoD requirements. Although the genesis of this document is the evaluation of COTS HR systems and their applicability to the DoD, the following recommendations are valid for the full spectrum of COTS applications, i.e. financials, payroll, manufacturing, inventory, etc.

**The DoD should assess their current investment in information technology for the HR domain. The integration of COTS HR with current development environments in DOD should be maximized in order to reduce cost and overall: time of implementation and leverage existing infrastructure investment.**

There should be an acknowledgment of the compelling business case for DOD, the Services and the taxpayer through the use of core COTS HR in conjunction with an open, non-proprietary, portable and widely used development environment (RDBMS & tools). Additionally, this business case should include documentation of the lower life cycle costs of ongoing efforts in "customized" COTS to support continuous improvement in an area of accelerating change and fiscal constraint. This assessment should be measured using the metrics of both dollars and time. With which vendors has the DoD obligated most of their funding? For a responsible vendor, a significant investment translates into a superior level of customer support. In which vendors products are the most developers and contractors trained? An established base of trained personnel keeps the learning curve shorter, allowing more time for productive system development. These factors weigh heavily in the likelihood of success.

**The DoD should select scalable, open systems components for their HR systems. One of these components must be an extensible HR product and a payroll product which can be seamlessly integrated at the DOD or service level, as well as proven implementation methodology and services capability.**

One of the keys to achieving success with the DoDs HR customers is the ability to effectively support the entire enterprise; from small workgroups to larger departmental organizations as well as providing agency level visibility. As we have seen, organizations are routinely downsizing and upsizing, commands are merging or dispersing. Scalability needs to be defined as both the ability for the system to seamlessly and rapidly increase in size or scale down based upon customer requirements. A rapid transition from an in-garrison configuration to a contingency operation should be achievable without diminished capability. Additionally, specific hardware, networking, operating systems, etc. should not be a gating factor in the scalability decision. The DoD should select open systems components which afford the agility to move to new technology with price/performance benefits without an onerous transition cost. Most current legacy systems deliver the functionality required at the time they were specified. However, infusing new technology into these systems is generally difficult and prohibitively expensive. DoD customers demand GUI applications, Web integration and other relatively new technologies from their systems.

**The DoD should identify a COTS HR provider who is able and willing to participate in a proactive partnership with DOD and has the corporate size, market experience and staying power to share in the development/integration of core and unique DOD military personnel system components.**

The COTS provider must consider the effort as equally important to their business as the DoD. They must have the corporate size, market share and commitment to share in the development effort. The COTS provider must also be able to assist in the migration of DOD and its components to best practices in personnel management/HR across national, industry and intergovernmental boundaries by affiliation with a global partner who provides the experience and customer base to achieve this critical objective.

**The DoD should initiate the expeditious development of service functional requirements in a standard repository which allows for easy comparison and rapid determination of common requirements. (extension of JWG process).**

The MPM 21 initiative should be used to drive towards the greatest degree of common requirements. The fewer service unique requirements that exist, the less duplicate development efforts will need to take place.

**The following are Oracle's perspective on the recommendations.**

**The DoD should assess their current investment in information technology for the HR domain...**

The DoD through their efforts with the Air Force at San Antonio, the Navy at New Orleans has spent millions dollars with Oracle. These procurements have been used to fund software, training and support. As a result, there is an Oracle HR knowledge base in DoD that far exceeds that of any other COTS HR provider. From a core technology perspective, Oracle maintains a 70+% market share of the relational database market within the federal government. This market share ensures that government requirements maintain high visibility within Oracle.

**The DoD should select scalable, open systems components for their HR systems...**

Oracle is portable and scaleable to a wide range of hardware environments and architectures. All DoD standard contract hardware and operating system platforms can run Oracle HR. Oracle is still the only vendor that can provide identical core technology from the desktop, through the workgroup server market up to and including enterprise level computing environments. Additionally, Oracle supports over 90 different hardware and operating system combinations as well as a wide variety of networking topologies. The transition from UNIX to NT to Windows is little more than an export and import of data. Oracle's Open Gateways and APIs provide an environment which over 3,500 vendors have used to develop products which integrate/interoperate with the Oracle product set. Oracle's Open Gateway technology allow for the transparent integration of non-Oracle data including; Sybase, Informix, DB2, VSAM, APPC, etc.

Oracle maintains an enviable list of industry firsts from the first commercially available SQL RDBMS to, most recently, web integration. This track record provides the DoD) with a high

confidence level that new technology will be integrated with the product, ensuring a leading edge end product for their HR customers.

**The DoD should identify a COTS HR provider who is able and willing to participate in a proactive partnership with DOD...**

Oracle has invested, and will continue to invest, millions of dollars in an effort to integrate government requirements into the core Oracle HR product. Through continual training, workshops, proofs of concept and briefings, Oracle, in a relatively short period of time, has demonstrated a tangible effort at partnership and joint accountability for the success of the COTS HR initiatives.

**The DoD should initiate the expeditious development of service functional requirements in a standard repository which allows for easy comparison and rapid determination of common requirements. (extension of JWG process).**

Oracle agrees with this initiative. Oracle plans to integrate DoD common requirements into the Oracle HR product. It is incumbent upon the DoD to agree to the maximum amount of commonality, thus insuring a greater level of COTS product capability.

**Mr. Larry Rinderknecht memo with EDS input to the  
Defense Science Board Task Force on Military  
Personnel Information Management Report - BAFO,  
(8/20/96)**



August 20, 1996

Defense Science Board Task Force Chairman and Executive: Secretary, Dr. Salisbury and Ms Norma St. Claire

From: EDS Military Systems

In July's meeting of the Defense Science Board (DSB) Task Force on Military Personnel, Dr. Salisbury offered the opportunity for everyone present to provide "Brief and Final Observations" on the recommendations to be made by the DSB Task Force.

EDS has been following the DSB deliberations since February when we made a presentation to the Board. As a large information systems integrator, we are and have been involved in many integration efforts similar to that being considered by the DSB. Of these, the one that is most comparable to the DoD in terms of scope and complexity is our integration work for General Motors (GM) in all functional areas including human resources and payroll. Those efforts began in 1984 and are continuing today.

Based on our GM experience, and that with other clients around the globe, we offer comments on three Terms of Reference (TOR) of the DSB.

**TOR 1: A single, fully integrated DoD personnel/payroll objective system for 2001 (MPM 21).**

We strongly recommend adoption of the **objective of having a fully, integrated DoD personnel and payroll system** by early in the 2000 decade.

When EDS was acquired by GM, and assumed responsibility for all of GM's IT personnel and assets, EDS was assigned the mission of supporting GM's production of world-class quality vehicles, while containing what had been ever increasing IT costs.

Accomplishing this mission in GM presented special challenges. The size of the bureaucracy, the fragmentation of efforts across different divisions, the outdated IT infrastructure of many organizations, the lack of technical talent possessing significant functional experience and the cost were factors that mitigated against an immediate integration solution. While many of GM's systems are fully integrated, others are not.

However in GM integrated systems always remained the objective, simply because of the payoff in large **resource savings and enhanced productivity**. Normally, the GM/EDS team worked first to establish "best practice" business processes and then to upgrade and fully interface GM's systems and infrastructure before tackling an integration solution. Payroll is a notable example as full integration is commencing just now.

Over the years we have found that **managing expectations was the singular most difficult task since integration is not easily, quickly or inexpensively achieved.**

We applaud the emerging DSB position of calling for a **common core as the right first step** to integrating the DoD's HR and payroll systems. Identification of a common core should begin with an in-depth process analysis which should result in identification of a common core of sufficient size to enable implementation of an integration solution in the time frame mentioned.

## **TOR 2: A COTS-based solution to generate savings.**

We believe a COTS-based solution is capable of generating savings for DoD, **provided careful limitations on scope of customizations are imposed.**

Our experience in working with many excellent COTS providers is that there simply is **no single COTS package that will totally fit** any corporate or government set of requirements. In the case of the DoD, this is particularly true if you include the unique characteristics of military manpower and tour assignments as part of the personnel function/system. Hence, **the reality for a DoD system is that some customization must occur.** The issue is how much? Customization of COTS applications cost money. If the customization is really extensive, it can make acceptance of subsequent upgrades of the COTS product prohibitively expensive.

To be worthwhile, and minimize cost, most purveyors of COTS solutions would tell you that a COTS package must meet two conditions in order to make a COTS solution a viable option for the organization to pursue:

1. As a generalized **rule of thumb, the package must have an inherent "fit/gap" of at least 80/20% to the current business processes, or**
2. **the "owners" of the business processes must be willing to tailor their processes to fit the COTS package.**

It should be recognized that customization is essentially a software "tailoring" effort that is an iterative process involving close work between vendor and customer. Even when done right, it is not a speedy process and may take months or even years to complete particularly if you include upgrades.

It is also worthy of note, that issues of ownership and maintenance of the "tailored" software quickly arise and need to be openly discussed and resolved up front. Finally, the ability of any COTS packages to handle high volume processing is a major issue that we faced in GM and should be evaluated for a very large customer like DoD.

## **TOR 3: A "generally sound" personnel community strategy that will rely on "Executive Agents" to implement the objective system for 2001.**

We believe that **use of service "Executive Agents" as change agents** to implement an objective DoD Personnel/Payroll system **can be a sound strategy** provided necessary management controls are in place to ensure continuous integration throughout the systems development/ implementation period.

Our experience in GM was that any process of IT integration/development required an "empowered" change agent (one having authority, responsibility **and financial control**). All lesser degrees of managerial control/coordination yielded deficient results. Ultimately the EDS/GM team adopted a new IT systems approval process to ensure continuous integration and continuous revalidation of business need throughout the development process. This process is still in use today.

We appreciate that operational and service needs may dictate a less centralized management solution within DoD. These can work, but will require much tighter integration of the interfaces between respective systems developed under the auspices of different "Executive Agents" or else, as GM chose, the assignment of such integration responsibilities to a third party. Our experience is that the requisite level of integration has to be much greater than the mere specification of standards or declaration of a generalized common operating environment.

EDS actively supports the efforts of the DSB Task Force and stands ready to assist the DoD and the Services as they move forward to defining and implementing the objective Military Personnel System 2001.

EDS appreciates the opportunity to share these thoughts with members of the Board. If we can provide any additional information, please call Deane Stanley or Larry Rinderknecht at (703) 742-1679 or 742-1651.

Respectfully submitted

***Military Systems Division  
13600 EDS Drive  
Herndon, Virginia 22701***